**二叉树非递归遍历-中序遍历**

一、

·延续我们上节的代码构造情况哈

·那么我们知道，其实递归是调用系统栈来进行操作，那么现在我们不用递归，也就是不利用系统栈来进行，所以我们只能够自己构造栈来模拟

·思想：

1. 我们先一直往左孩子走，边走边入栈，直到走到为空，我们pop栈顶元素并且打印出来
2. 我们访问打印出来的元素的右孩子，然后继续循环上面的操作
3. 就这样，那么我们退出的条件是栈为空且当前访问的元素也为空就可以退出了
4. 代码

//中序遍历，其实就是在先序遍历的基础上，把输出的函数放在中间而已

void ind\_tree(BiTree\* T) {

if (!T) return; //如果为空就返回

ind\_tree(T->left\_child);

cout << T->data << " ";

ind\_tree(T->right\_child);

}

#include <stack>

//非递归中序遍历

void noRec\_ind\_tree(BiTree\* T) {

stack<BiTree\*> S;

BiTree\* p = T;

do {

//现在我们先一直往做走

while (p) {

S.push(p);

p = p->left\_child;

}

//走到尽头了，我们现在先出栈一个，再去访问其右孩子

BiTree\* cur\_p = S.top();

S.pop();

std::cout << cur\_p->data << " ";

p = cur\_p->right\_child; //注意这边我写的是当前出栈的元素的右孩子哦

} while (!S.empty() || p); //退出的条件哦

}

运行截图

